SQL-first application development with *sqlc*

Nikolay Kuznetsov

Rapperswil, 26 June 2026

About me

- Senior Software Engineer
- Zalando Helsinki
- *sqlc* contributor, *Docker* captain
- Author of *pgx-outbox* and *sqlc++* projects


sqlc

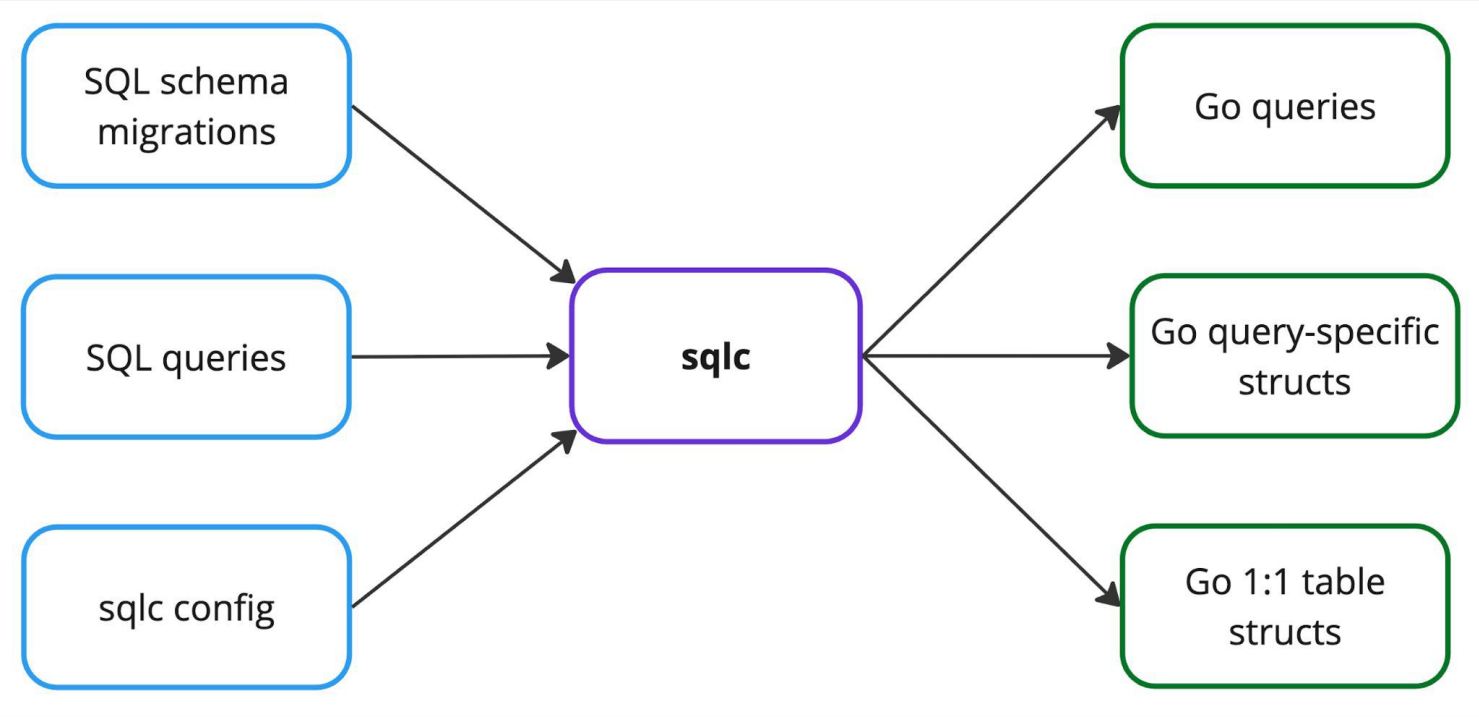
sqlc intro

- Static code generator
  - **Go**, Typescript, Python, Kotlin
- SQL-first approach
- Schema-aware
- Compile-time type safety

Postgres from Go options

- ORMs (GORM, Ent, Bun)
- **pgx** driver and *pgxpool* API
- *squirrel* to build SQL queries
- **sqlc -** SQL compiler

sqlc in a nutshell



Cart items table

```
CREATE TABLE IF NOT EXISTS cart_items (

    owner_id        VARCHAR(255)                                NOT NULL,
    product_id      UUID                                        NOT NULL,
    price_amount    DECIMAL                                     NOT NULL,
    price_currency  VARCHAR(3)                                  NOT NULL,

    created_at      TIMESTAMP DEFAULT CURRENT_TIMESTAMP NOT NULL,

    PRIMARY KEY (owner_id, product_id)
);
```

Get cart query

```
-- name: GetCart :many
SELECT product_id, created_at
FROM cart_items
WHERE owner_id = $1
```

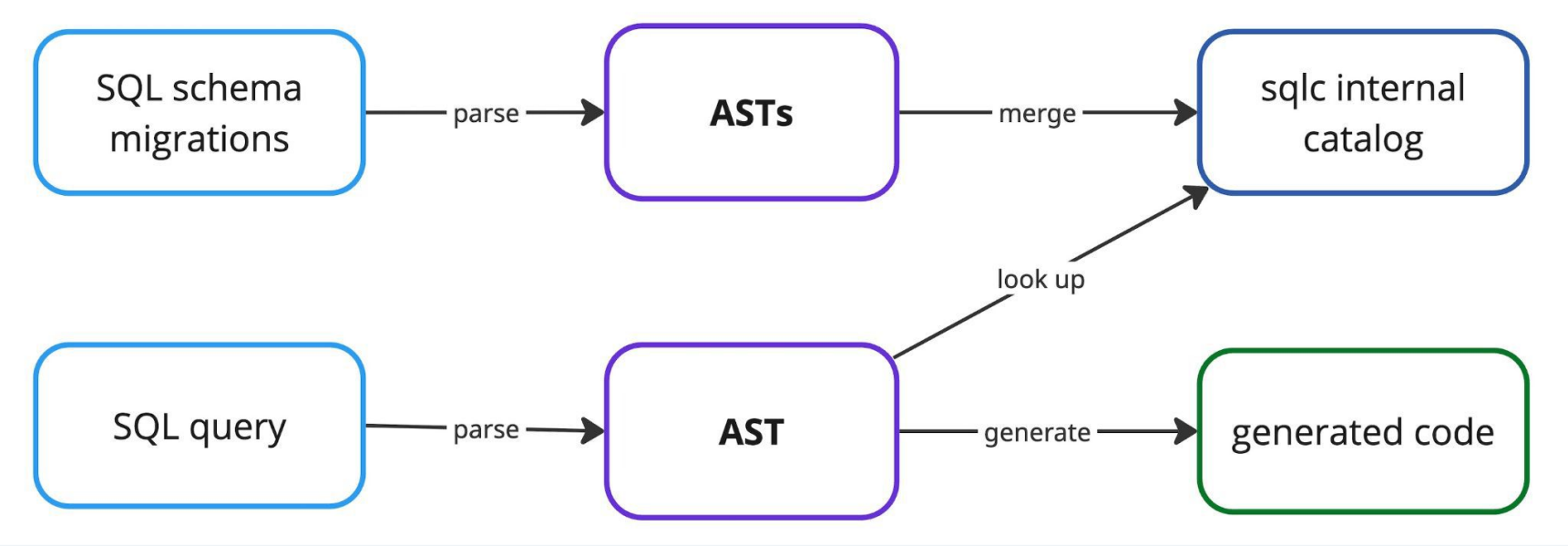
More options: exec, one, **many**, batchXYZ, etc

Generated record and query

```go
// Code generated by sqlc. DO NOT EDIT.
//   sqlc v1.31.1

type GetCartRow struct {
    ProductID string
    CreatedAt time.Time
}

func (q *Queries) GetCart(ctx, ownerID string) ([]GetCartRow, error) {
    rows, _ := q.db.Query(ctx, "SELECT ...", ownerID)
    defer rows.Close()

    var items []GetCartRow
    for rows.Next() {
        var i GetCartRow
        _ = rows.Scan(&i.ProductID, &i.CreatedAt)
        items = append(items, i)
    }

    return items, nil
}
```

sqlc under the hood

In-memory mode:



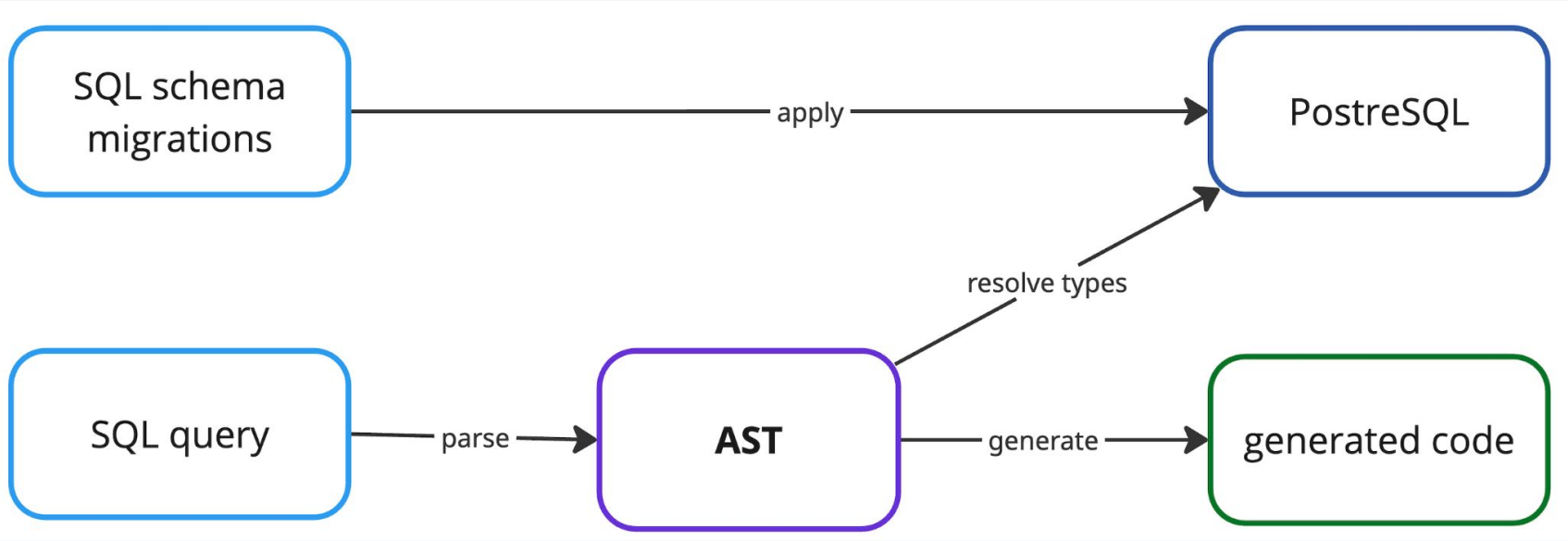
Simplified Abstract Syntax Tree

```
-- name: DeleteItem :execrows
DELETE FROM cart_items WHERE owner_id = $1 AND product_id = $2;
```

```
DeleteStmt
  ├ Relation: cart_items
  └ WhereClause (BoolExpr: AND)
      ├ left: ColumnRef { Name: "owner_id" } = ParamRef{1}
      └ right: ColumnRef { Name: "product_id" } = ParamRef{2}
```

sqlc under the hood

DB mode:



Pros of *sqlc*

- Less Go boilerplate code to write
- Compile-time schema and type safety
- Separation of SQL and Go code

Thank you!

Nikolay Kuznetsov


nikolayk812 / sqlcpp


nkuznetsov